INTEGRATED COMPUTER CONTROL SYSTEM

P. VanArsdall
R. Bettenhausen
F. Holloway
M. Miller
B. Reed
J. Spann
J. Wiedwald

During Title I, we defined the general software approach and hardware systems for NIF's integrated computer control system (ICCS). The ICCS design incorporates CORBA—common object request broker architecture—into a distributed, client–server network. The ICCS is a layered architecture consisting of supervisory systems that coordinate front-end processors. The supervisory systems provide centralized operating controls and status for laser systems such as the Pockels cell, alignment controls, and optical pulse generation. The supervisory system also handles data archiving and integration services. Front-end processors provide the distributed services needed to operate the approximately 36,000 control points in the NIF. During Title II, we will refine our design and begin the actual software coding, an activity that will continue throughout Title III.

Introduction

NIF's complex operation, alignment, and diagnostic functions will be controlled and orchestrated using the ICCS. The ICCS must integrate about 36,000 control points, operate continuously around the clock, and be highly automated and robust. The system architecture also must be flexible; typically, control systems for complex facilities such as the NIF must be able to absorb significant changes in requirements late in project construction. To mitigate risks to the infrastructure, the ICCS design incorporates modularity, segmentation, and open systems standards. This design allows components and subsystems to be replaced at designated interface points, if necessary. Risks to the control system software are managed through a modern, object-oriented software framework that is used to construct all the applications. This framework is extendable and maintainable throughout the project lifecycle. This framework also offers interoperability among computers and operating systems by leveraging CORBA—a "common object request broker architecture" (see "What Is CORBA?" on facing page).

During NIF's 30-year lifetime, the operation of the facility will evolve and computer technology will change. We are planning for this evolution by adhering to existing industry standards, such as UNIX for the operating system; a structured query language (SQL) for the database technology; Ada for the programming language; X-Windows and Motif for the user interface; and CORBA for the distributed software.

In this article, we summarize the ICCS's general architecture, the computer system, the supervisory software system, the application front-end processors, the integrated timing system, and the industrial controls system, as well as the integrated safety system.

System Architecture

The ICCS architecture was created to address the general problem of providing distributed controls for a large-scale scientific facility that does not require significant real-time capability within the supervisory software. Figure 1 shows a simple view of the entire NIF computer system. The ICCS is a layered architecture with a supervisory system (i.e., the upper-level computers) controlling the front-end processors (FEPs). The supervisory layer, which is hosted on UNIX workstations, provides centralized operator controls and status, data archiving, and integration services. The FEPs are constructed from VME-bus or VXI-bus crates of embedded controllers and interfaces that attach to the control points. FEP software provides the distributed services needed by the supervisory system to operate the control points.

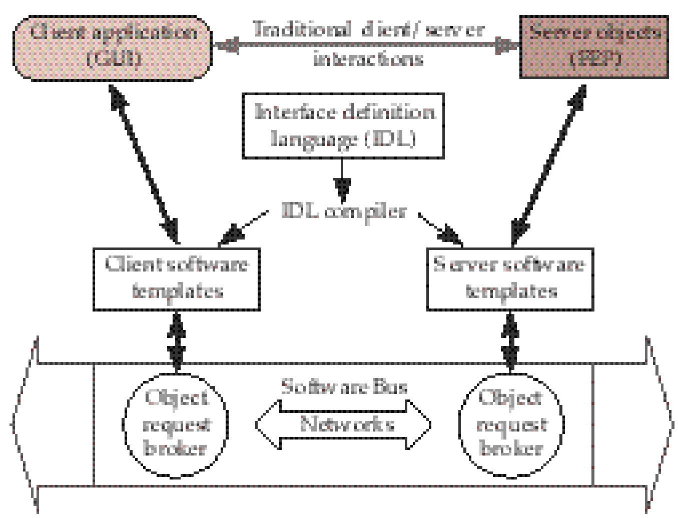
Eight supervisory and 13 FEP applications are used to implement the specific requirements of the NIF control system, as shown in Figure 2. The topmost layer is the Shot Director supervisory application, which coordinates the supervisory subsystems to provide shot integration.

WHAT IS CORBA?

CORBA is a standard developed by a consortium of major computer vendors (The Object Management Group) that addresses the need for interoperability among hardware and software products. The best way to think of CORBA is as the universal “software bus.” CORBA is a series of sophisticated, but standard sockets into which software objects can “plug and play” to interoperate with one another. Even when made by different vendors at different times, the object interfaces are standard enough to coexist and interoperate. By design, CORBA objects operate across languages, operating systems, networks, and tools.

When objects interact, it is convenient to label one “the client” (that is, the object that initiates the interaction), and the other “the server” (the object that responds to the initiative). CORBA provides tools for building both clients and servers, and allows an object to be a client in some interactions and a server in others. Historically, the interface between clients and servers were separately defined and developed for each application, type of machine, and computing environment. It was unusual to find much in common. By meeting the interface definition of CORBA, the clients and servers of applications can now easily communicate with one another.

The figure shows a greatly simplified diagram of CORBA’s major parts. The object request broker (the “ORB” in CORBA) establishes the client–server relationships between objects. Using an ORB, a client can invoke a method on a server object. The ORB intercepts the call and is responsible for finding an object that can implement the request, pass to it the parameters, invoke its method, and return the results. The interface types and methods provided by the servers and used by the clients are defined by an industry-standard Interface Definition Language (IDL). The IDL compiler examines the interface specification and generates the necessary interface code and templates into which user-specific code is added. The code in the client that makes use of CORBA objects is written as if the server were locally available and directly callable— CORBA takes care of all the rest.



CORBA “brokers” between client and server software systems using objects as the common components. In traditional client/server interactions, the client must be programmed to send requests directly to the server. (40-00-0298-0341pb01)

Seven other supervisory applications provide operator control and status. In the next level, application FEPs introduce the capability for autonomous control, provide services to the upper layers, and use the services from the layer below. These application FEPs control ten applications: wavefront, power conditioning, laser energy, laser power, master oscillator room (MOR), automatic alignment, preamplifier module, plasma-electrode Pockels cells (PEPCs), target diagnostics, and industrial controls. The bottom layer comprises FEPs that provide device control services to the upper layers. These server FEPs control three “server” applications: alignment controls, timing, and video.

This architecture is generally applicable to event-driven control systems where client–server tactics are appropriate, as in the case of NIF. For NIF, the shot timeline occurs over several hours and can be suspended if necessary. Some real-time control is necessary; this is handled by the specific subsystems, such as the Integrated Timing System (see p. 208), or at the edges of the architecture. The ICCS client–server architecture will meet NIF’s shot timeline requirements, as shown in Figure 3.

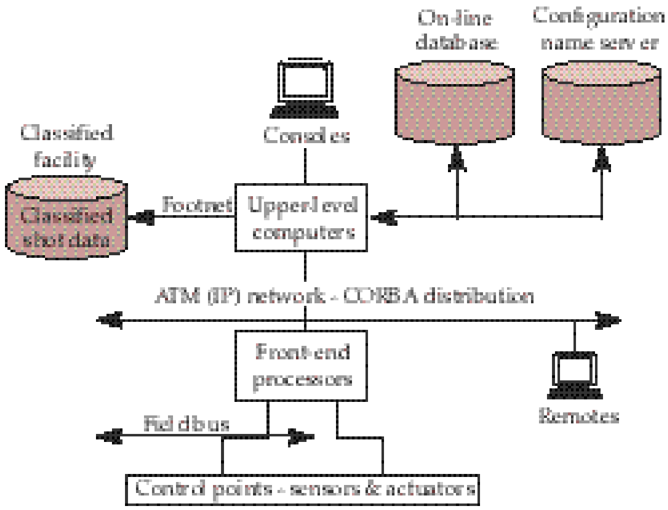


FIGURE 1. The NIF computer system includes upper-level (supervisory) computers, which are connected to consoles in the NIF control room, front-end processors, and remote consoles in the NIF laser bay and target area. The system also includes an on-line database, a configuration name server, and a classified facility for storing classified shot data. (40-00-0298-0323pb01)

Title II Activities

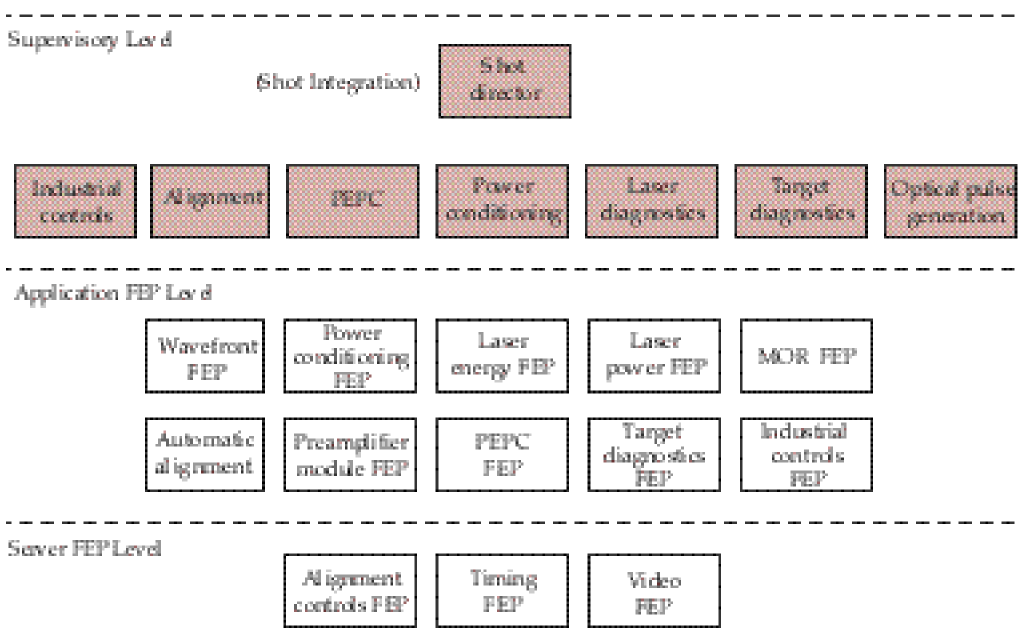
During Title II, we will complete hardware designs and propose plans for procurement, assembly, and installation. Simulation of key parts of the control system will be conducted to ensure adequate capacity and performance of the ICCS infrastructure. We are taking an iterative approach to software development. Our general strategy is to deliver the most needed functionality first, adapt to changing requirements when they become known, and add increasing detail in later releases. During Title II, we plan to activate a production prototype of the software, a year before the first NIF bundle. The prototype will demonstrate the capability of this software design to meet key NIF requirements. Most of the detailed software coding will be done during Title III.

Computer System

The NIF computer system can be divided into two broad areas: the software engineering and the operations computer systems. The software engineering computer system includes software development tools, development and testbed computers, and software targets. The operations computer system includes file servers and run-time commercial software, control room and remote operator consoles, and the graphical user interface (GUI).

The software engineering computer system will be used to develop software for the facility (see “Developing NIF Software” facing page). This system supports both supervisory computers and FEPs. Most FEP targets are PowerPCs. For the supervisory applications, we picked the Solaris operating system. The hardware includes a Sun Ultra-SPARC file server, “Sun SPARC 5” workstations, a switched Ethernet local area network, and an asynchronous transfer mode (ATM) switch. The software development tools include a Rose object-oriented design

FIGURE 2. NIF supervisory and front-end processor software subsystems. (40-00-0298-0324pb01)



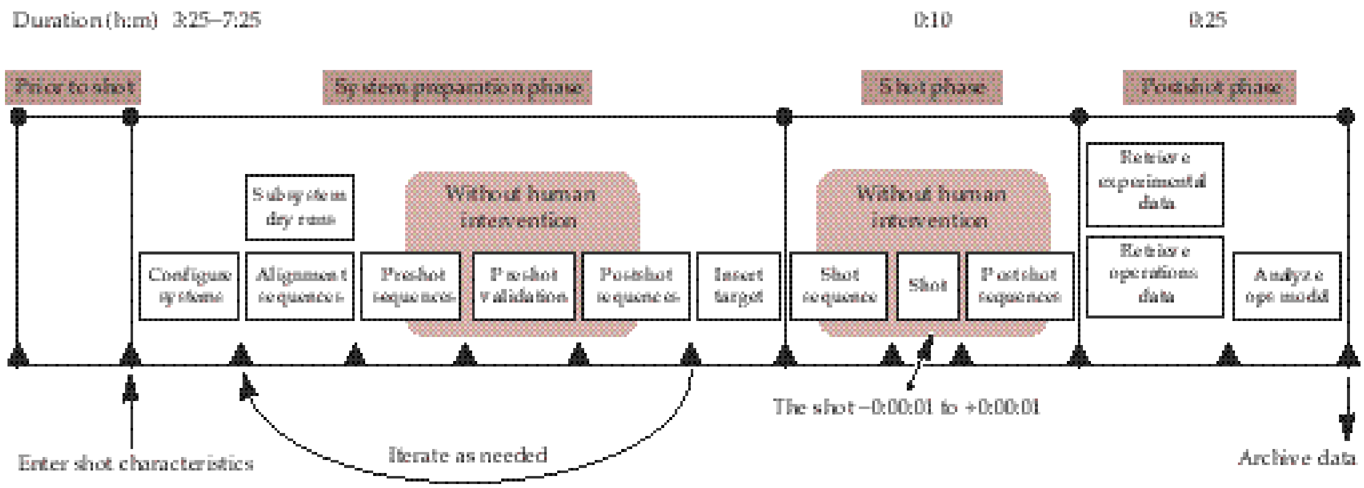


FIGURE 3. The shot timeline. Because the sequence can hold indefinitely, by iterating through the system preparation phase, event-driven techniques are appropriate. A "real-time" NIF shot lasts about 2 s. Other operating modes include computer start-up, maintenance, and commissioning. (40-00-0298-0325pb01)

tool, Ada 95 and C++ integrated compilers, and an Apex development environment. Other software tools include Objective Interface Systems CORBA for Ada 95, a UIM / Orbix graphical user interface tool, and an Oracle relational and object-oriented database. For the FEP targets, we use VxWorks Tornado real-time UNIX. Figure 4 shows an early Title II design for the software engineering computer system.

Figure 5 shows the Title I operations computer system (i.e., the computers that control the NIF) and network. In the computer room, a pair of file servers provides disk storage and archival databases for the entire system. The file servers also host centralized management and naming services necessary for coordinating the facility operation. Each server has two central processing units (CPUs), expandable to four, and 512 Mbytes of memory, expandable to 30 Gbytes. Servers also have hot-swappable components, redundant power, and redundant cooling. The storage array has 75 Gbytes of memory, expandable to 300 Gbytes, and hot plug disk drives.

DEVELOPING NIF SOFTWARE

ICCS software development is managed under a software quality assurance plan that covers the entire life cycle of the software design, production, and maintenance. Central to the software development are documentation standards for requirement specifications (SRSs) and software design descriptions (SDDs). These documents are essential to the long-term maintainability of the software in view of the periodic software upgrades and staffing turnover expected over the 30-y life of the NIF.

First, we analyze the SRSs and use the Rose tool to obtain object-oriented designs. We then examine classes, or sets of objects that share common structures and common behaviors, to find any abstractions and patterns. Rose then generates code specifications in Ada and Interface Definition Language (IDL)—these specifications form the design description for the software. The software engineers write detailed codes, according to the code specifications. The resulting code is compiled, linked, and debugged. If the code is for supervisory applications, it is compiled by an Apex Ada 95 self-compiler and linked for the Solaris target. If the code is for an FEP, it is compiled through an Ada 95 cross-compiler, before linking to a Power PC VME target (or others) existing in a VxWorks Tornado real-time environment. The working software is finally reverse-engineered to capture any specification changes back into the original Rose model.

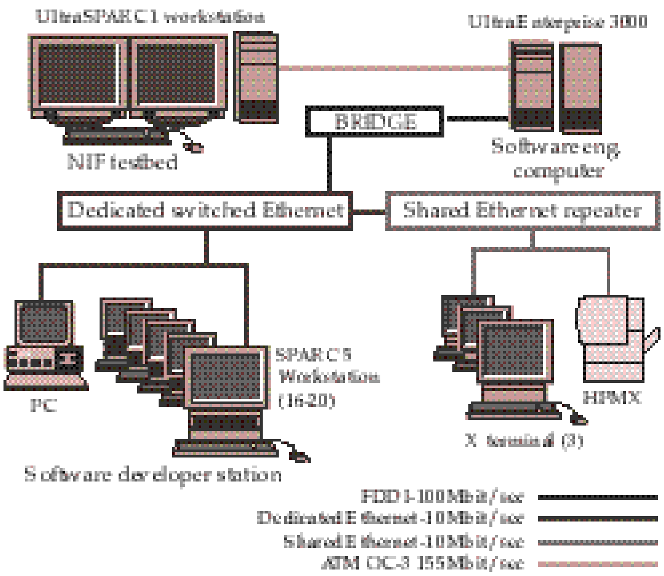


FIGURE 4. Software engineering computer system layout. (40-00-0298-0326pb01)

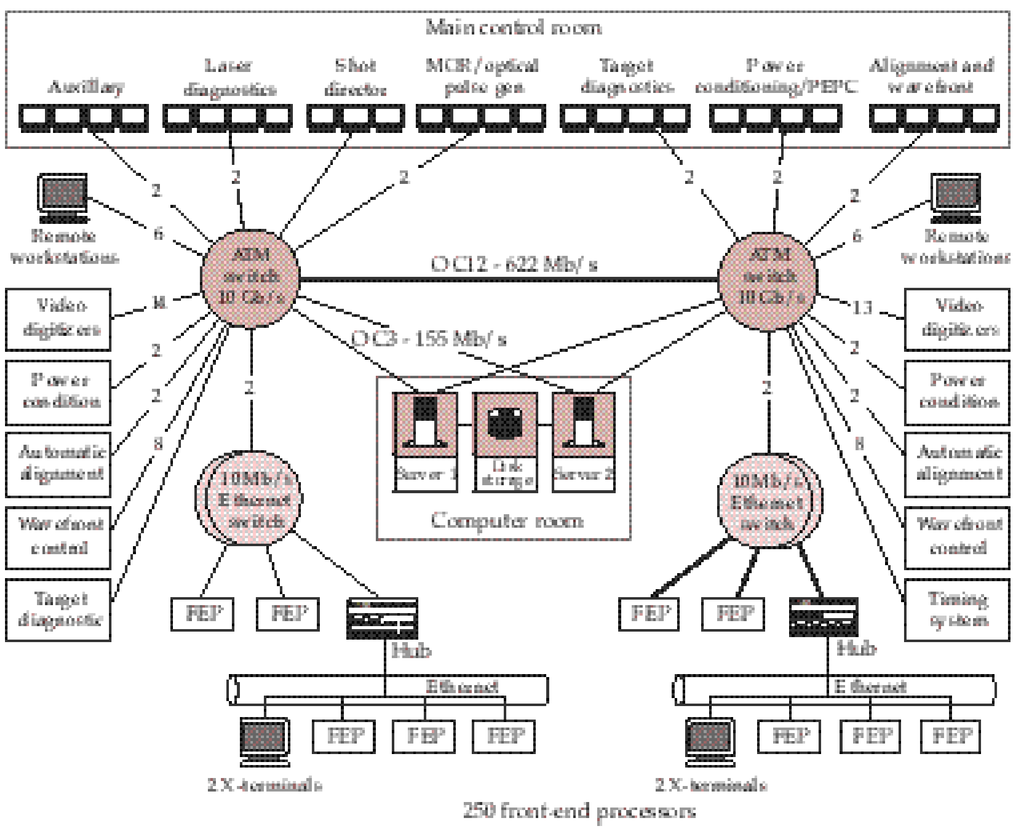
The network backbone is built from ATM switches capable of carrying digital motion video as well as standard Internet protocol (TCP/IP). TCP/IP is used for most communications. FEPs are generally attached to shared Ethernet, switched Ethernet, or Fast Ethernet depending on the bandwidth requirements of the control devices and supervisory communication. Some FEPs and all workstations and file servers are attached directly to ATM for maximum communications performance.

Figure 6 shows the main control room layout. The seven graphic consoles include one dedicated to the shot director and six operator consoles. The shot director's console has three color monitors and an UltraSPARC 1 with 256 Mbytes of memory. Each operator console has two workstations with two color monitors per workstation. Figure 7 shows where the remote consoles will be located in the facility. There are 13 video-capable consoles and four X-terminal consoles.

Title II Activities

Early on in Title II, we will procure and activate the remaining software engineering computer tools and components. We will also support the first phase of software framework construction. This construction includes demonstrating the integration of vendor-supplied

FIGURE 5. NIF computer system and network (40-00-0298-0327pb01)



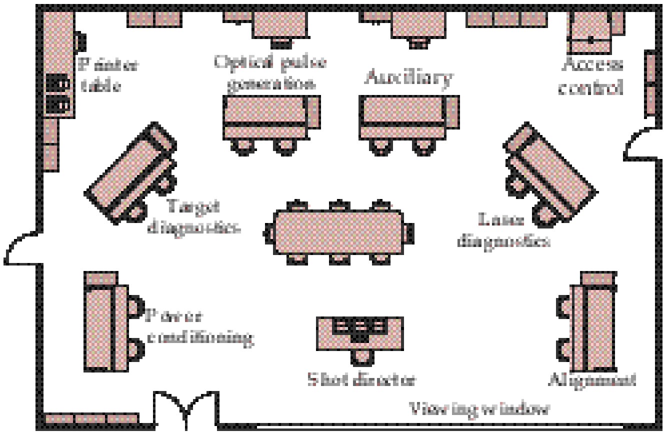


FIGURE 6. Operator control room layout. (40-00-0298-0328pb01)

software tools, improving our understanding of CORBA performance scaling, and constructing a sample FEP to demonstrate how the tools work. During Title II, we will also develop a more detailed model for guiding computer sizing and software deployment.

Supervisory Software System

The supervisory controls must provide semiautomated sequencing of NIF shots, GUI operator controls, event-driven status reporting for broad-view status displays, shot-data processing, reporting, and archiving, and time-stamped logging and abnormal event ("exception") handling.

The ICCS supervisory software provides integrated control for NIF's seven supervisory applications—alignment controls, the PEPC, power conditioning, target diagnostics, optical pulse generation, laser diagnostics, and industrial controls. It also provides manual operator controls for maintenance; acquires, displays, archives, and reports laser and target-area shot data; configures laser and target-area sensors using FEP capabilities; and provides control room interface for facility environmental monitor, access control, and safety interlocks.

We derived the performance requirements for the supervisory systems from the operators' needs for timely information and interactive responses. The broad-view status update must complete in 10 s, and some GUIs require 10 updates / s. The software is event-driven: status information is propagated from the laser to updates on the graphic user screens. Some process controls are encapsulated in the FEPs and are not part of the supervisory system, notably wavefront control, automatic alignment, and capacitor charging.

The supervisory system is divided into application software—including a database management system—and frameworks, as follows.

Application Software

Application systems perform NIF functions—they execute operator commands, report on machine status, and manage operational and archival data. These supervisory applications play major roles in the life cycle of a NIF shot. While setting up a plan

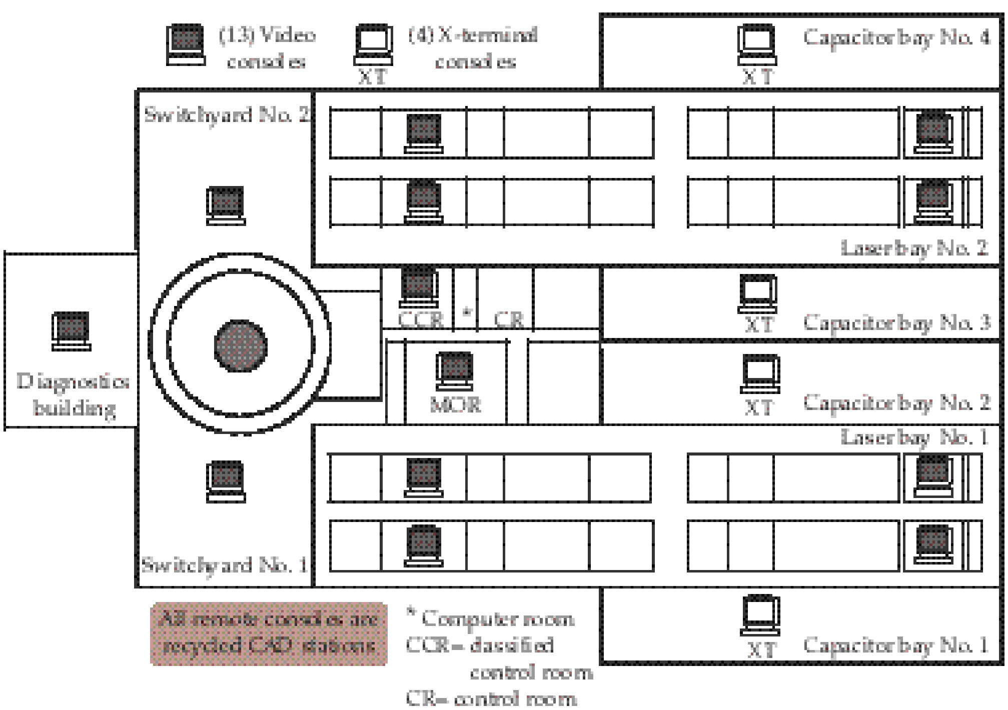


FIGURE 7. Location of the remote consoles in NIF. Graphical user interface controls and motion video are available through these consoles. (40-00-0298-0329pb01)

for a “system shot” (a shot when the laser is fired to satisfy a set of experiments), experiments may be added or removed, or experimental goals may be changed. There are also nonsystem shots, including the preamplifier module shot, the target alignment sensor shot, and the dry-run shot. As a result of these nonsystem shots, some setup plan parameters are adjusted. Finally, when the system shot is taken, the results are archived and the shot is complete. During this life cycle, there are six ICCS software control phases, as follows:

- Phase I—pick the experiments that will make up the shot.
- Phase II—derive values for the laser hardware to accomplish the shot, perform laser model calculation, and refine the experiments.
- Phase III—set the derived values in the laser hardware to accomplish the shot.
- Phase IV—ensure that the hardware is properly set up and disable changes.
- Phase V—perform the final, time-critical setup, and pass control to the integrated timing system.
- Phase VI—save the information for later analysis.

Figure 8 shows approximately where these phases occur in the timeline of a NIF shot. In “real time,” the NIF shot lasts about 2 seconds.

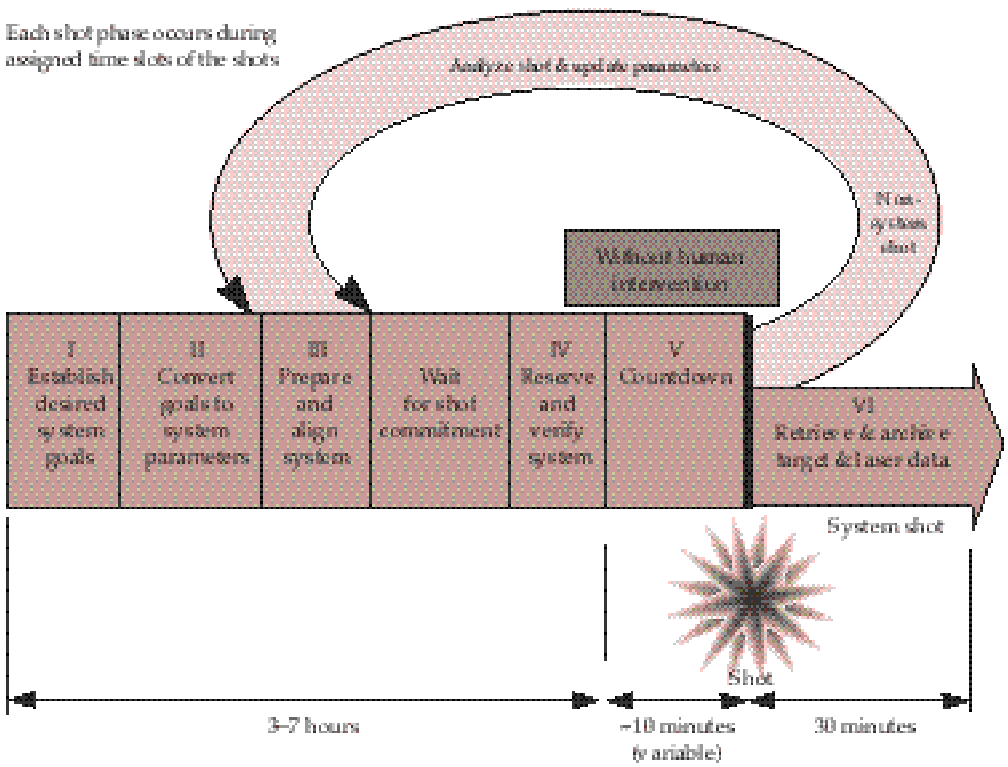
Database Management System

The supervisory applications will draw upon and create an enormous amount of data that must be stored and archived. To manage this data, we are designing an object-relational database management system. This database system will include:

- A configuration database, which stores persistent information pertaining to control system operation, such as device parameters.
- Experiment data sheets, which store experiment goals. Each NIF shot may involve many sheets.
- The shot setup plan, which stores parameters that describe setup conditions for executing a shot. The plan is the collective goals of all experiment data sheets for a particular shot.
- The shot data archive, which temporarily stores data associated with a shot. Each subsystem will likely have its own archive application program, and a record of what is put into each archive package is stored in the database.
- Machine history, which will store the history for each NIF component of interest (e.g., motors, photodiodes, cameras, switches). This history will be used to characterize performance, schedule preventive maintenance, trigger calibration, etc.

For our database design, we are assuming that we will keep online one year’s worth of experiment data

FIGURE 8. Estimate of where the six ICCS software control phases will occur in the timeline of a NIF shot. This figure also shows that setup parameters are readjusted after a nonsystem shot. (40-00-0298-0330pb01)



sheets and shot setup plans, 20 shots' worth of shot archive data, machine history for 100,000 components, two weeks' worth of optics inspection data, and configuration data for 50,000 components. We are estimating that we will need a total of 1278 database tables, and about 66.7 Gbytes of disk space.

Frameworks

The ICCS supervisory software framework is a collection of collaborating abstractions used to construct the application software. A framework reduces the amount of coding needed by providing prebuilt components that can be extended to accommodate specific additional requirements. A framework also promotes code reuse by providing a standard model and interconnecting backplane that is shared from one application to the next.

Components in the ICCS framework plug into the CORBA bus. The ten frameworks that form the basis of the ICCS supervisory software are as follows.

The *configuration framework* provides a hierarchical organization for the static data that define the hardware control points accessible to the ICCS. Configuration provides a taxonomic or hierarchical "naming" system that is used as the key by which clients locate devices and other software services on the CORBA bus.

The *status monitor framework* provides generalized services for broad-view operator display of device status information. The status monitor observes devices and notifies other parts of the system when the status changes by a significant amount.

The *sequence control language (SCL) framework* is used to create custom scripting languages for the NIF applications. The SCL service automates sequences of commands executed on the distributed control points or other software artifacts.

The *GUI framework* ensures that the GUIs displayed upon the control room consoles or X terminals are consistent across the applications. The GUI framework is based upon the X Windows system and Motif policies.

The *message log framework* provides event notification and archiving services to all subsystems or clients within the ICCS. A central server collects incoming messages and associated attributes from processes on the network, writes them to appropriate persistent stores, and also forwards copies to interested observers (primarily GUI windows on the screens of operators' consoles).

The *reservation framework* manages access to devices by giving one client exclusive rights to control or otherwise alter the device. The framework uses a lock-and-key model; reserved devices that are "locked" can only be manipulated if and when a client presents the "key." Read access to obtain status is not affected by the reservation.

The *system manager framework* provides services essential for the integrated management of the hundreds of computers on the ICCS network. This framework ensures that necessary processes and computers are operating and communicating. Services include parameterized system start-up, shutdown, and process watchdog monitoring.

The *machine history archive framework* collects information that originates within the ICCS about the performance and operation of the NIF. This data is used in analyzing the NIF operation to improve efficiency and reliability.

The *generic FEP framework* pulls together the distributed aspects of the other frameworks (in particular, system manager, configuration, status monitor, and reservation) by adding unique "classes" for supporting device and controller interfacing. The generic FEP also defines a common hardware basis including the target processor architecture, backplane, I/O boards, device drivers, and field-bus support. (See "Front-end Processors" below for more information about the generic FEP.)

The *shot data archive framework* allows collecting the data from the diagnostics, making the data immediately available for "quick look" analysis, and delivering the data to an archive. The framework contains a server working with the system manager to assure that requested shot data are delivered to a disk staging area. The archive server is responsible for building a table-of-contents file and then forwarding the table and all data files to the archive.

Title II Activities

During Title II, we will begin designing detailed object models of substantial portions of the frameworks for the supervisory software. We will use the ICCS testbed to demonstrate control integration in FY98, and deliver basic supervisory software services when the first NIF bundle is activated in FY00.

Front-end Processors

We currently estimate that NIF will have over 293 FEPs to integrate control points such as sensors and actuators. These control points attach to interface boards plugged into the FEP backplane. In many cases, control points are handled by intelligent components such as stepping motor controllers, photodiode detectors, and power supply controls. These components incorporate local microprocessors operated by small fixed programs (called embedded controllers). In a few cases, remote devices attach to FEP units by using a low-cost network of microcontrollers known as a "field bus."

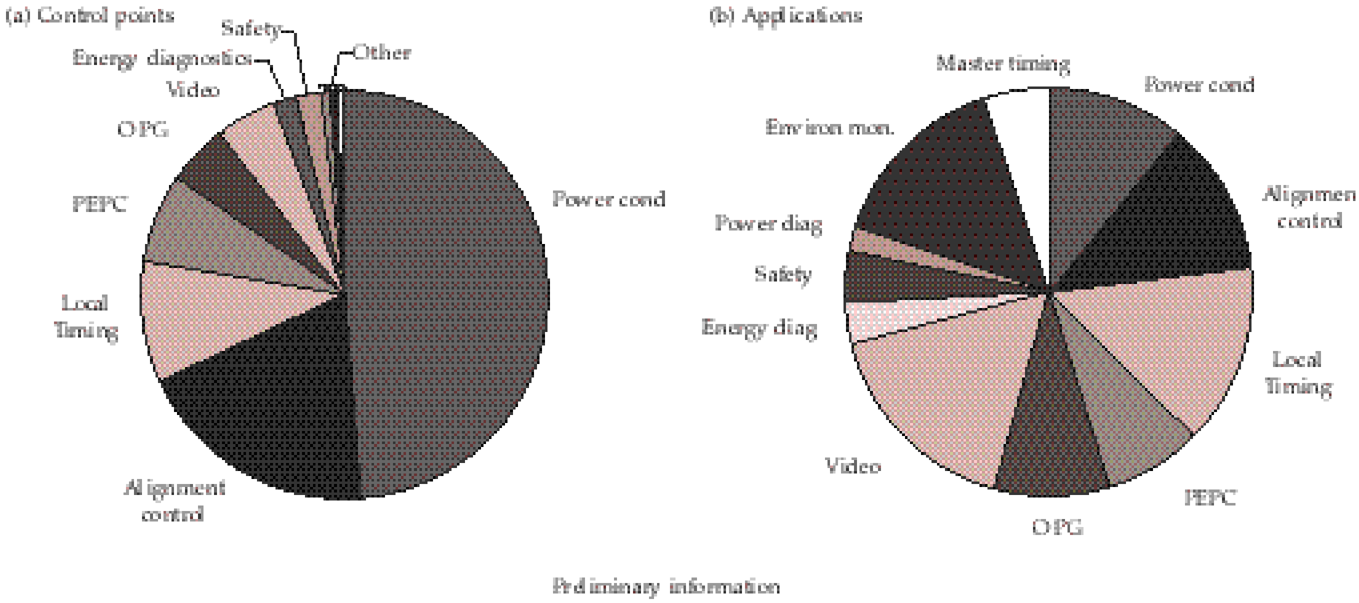


FIGURE 9. Charts showing relative amounts of (a) control points and (b) applications for the various NIF systems (preliminary information, based on Title I design). The present design has 35,826 control points for 96 kinds of devices. These device types are used in 461 different applications. (40-00-0298-0331pb01)

Every control point or device has a unique control-system name, called a “taxon.” The elements of that taxon identify where the device is in the NIF system (i.e., what major system it resides on and the “area” within that system), the equipment that contains it [i.e., the container—sometimes a line-replaceable unit (LRU)—and the package within the container], and what kind of device it is (whether a gimbal, shutter, etc., and which particular gimbal or shutter). For Title I, our preliminary information shows that there are about 36,000 control points of 96 kinds of devices in 461 applications (see Figure 9).

Our goal is to build the FEPs with as many common elements as possible to reduce software development and risk. To that end, we are developing a generic FEP, which we will provide to the FEP designers. Figure 10 shows the contents of a generic FEP. It will include frameworks (those software codes that call up services common to all FEPs); a common design; common tools and operating environment; common start-up and control methods; common hardware, where possible; and common test software. At the top of Figure 10 are the frameworks including the user interface, the message log and alert, and the system manager. Each FEP will also contain code to support the device reservation and status monitor frameworks. There are also specific FEP applications. Each FEP will include a device simulation code that allows a supervisory-level system to simulate that FEP’s activity, if needed. Each FEP will also have a real-time operating system and drivers that go out to the hardware devices and echo back. For an example of a specific FEP design, see “Automatic Alignment Front-end Processor” on facing page.

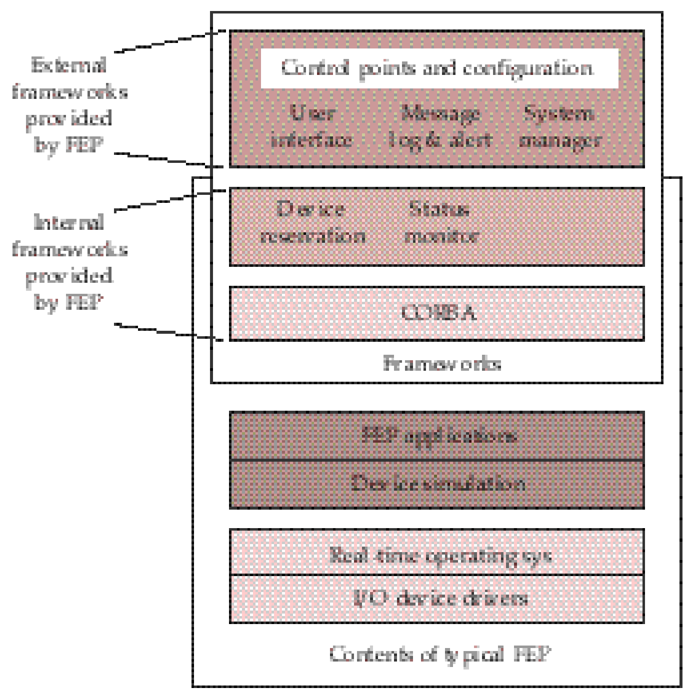


FIGURE 10. A generic front-end processor. (40-00-0298-0332pb01)

Title II Activities

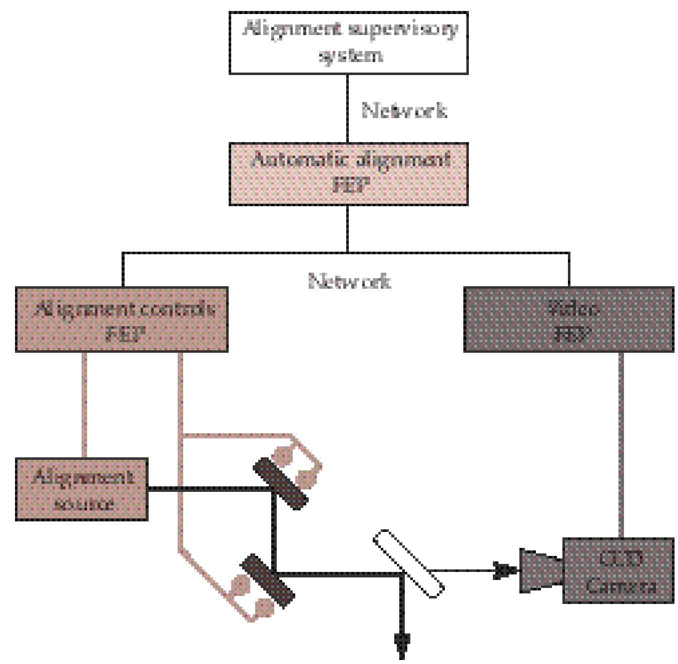
During Title II, we will establish uniform terminology and control point taxons throughout the design materials and drawings and will complete input and addition of information on the control

AUTOMATIC ALIGNMENT FRONT-END PROCESSOR

The requirements imposed on the FEPs for automatic alignment reflect the requirements on the alignment system. This system must automatically align NIF within 30 minutes, point the beam to align with the pinholes, center the beam to align apertures of several optical sections, adjust the beam's 3D orientation, focus the beam onto the target, adjust the KDP angle to match beam pointing, and balance the spatial power distribution of the serrated aperture.

As a result of these requirements, the FEPs have many functions. The primary function is to align the beamlines from the front end of NIF to the target. With more than 3000 devices to align, timely alignment would be impossible without an automated system. The automatic alignment software system must perform electro-optical characterizations, making measurements with video images; determine cross-coupling of some groups of devices; and determine pixel (image) to steps (motors) scale factors. Finally, it must accommodate classified targets, using a classified FEP to align the beamlines. An actuator control translator provides the required control information from the classified computer to the unclassified devices.

To accomplish these tasks, the automatic alignment FEPs interact with the alignment controls supervisor and laser diagnostics supervisor, as well as the service-level FEPs for video and alignment controls. The generic control loop used to align a single beam is shown in the figure at right. There are four basic types of alignment control loops—pointing, centering, rotation, and focusing. The pointing loop must compare beam images with reference reticles; the centering loop compares centroids from two pairs of light sources; the rotation loop compares orientation of two pairs of light sources; and the focusing loop adjusts the spot size of the beam on the target.



In the generic control loop, the automatic alignment FEP accepts alignment sequence commands from the supervisory system. The FEP sets up the beamline, commanding the alignment controls FEP to select the source, set the shutters, set filters, etc. It requests and acquires a video image from the video FEP, processes the image, and determines the alignment error by comparing the alignment attributes from the image with a stored reference. It adjusts the component by commanding the alignment controls FEP to manipulate optical devices. Finally, the automatic alignment FEP iterates by reacquiring the image, determining alignment error, etc., until the measurement is within tolerance. Over 3000 of these loops must be closed before each shot. (40-00-1298-0343pb01)

Our Title I hardware design is based on Sun's Enterprise 3000 system, which uses Ultra-SPARC processors. To meet the performance requirements, we will have four automatic alignment FEPs to align the 192 beamlines. Alignment will require 17 control loops per beamline: 8 for pointing, 6 for centering, 2 for rotation, and 1 for focusing. We anticipate four iterations per control loop to reach tolerance.

devices. We will also identify and assign taxons to the electronic equipment, computers, networks, and computer interfaces as well. We will review the software catalog for common approaches or any unnecessary devices, and initialize the structure of the configuration database. During Title II, we will also complete the design of a generic FEP, and build the first release, which will support the Title II-level frameworks discussed in the section on supervisory software. Finally, we will deliver this version of a generic FEP to the FEP designers, who are developing FEPs for the various NIF systems.

Integrated Timing System

The integrated timing system (ITS) provides triggering and timing signals to several client systems: the optical pulse generation system, the power conditioning system, the amplifier flashlamps, the PEPCs, laser diagnostics, and target diagnostics. The ITS has specific operational and performance requirements. Operationally, it must provide client control of all relevant parameters, support activation and maintenance activities by clients, quickly detect and recover from a hardware failure, and remain functional over NIF's 30-year lifetime. For performance, it must supply 1200 triggers while satisfying clients' range and stability requirements. We are providing three ITS performance levels to provide timing signals (Figure 11).

ITS has three subsystems: cross timing (fiducial distribution), local timing distribution, and facility timing distribution (Figure 12). The facility timing distribution system uses a standard, two-way time-transfer technique

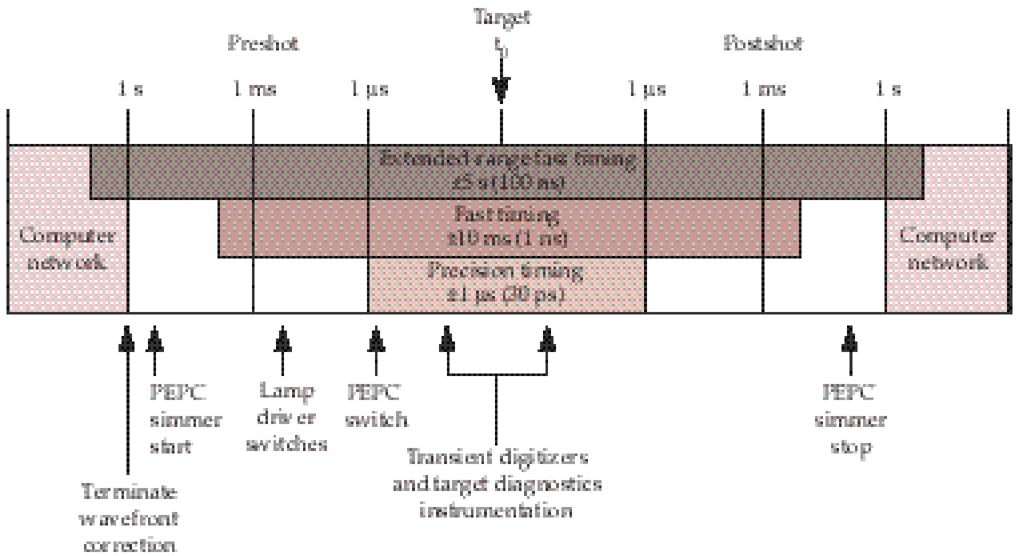


FIGURE 11. Three ITS performance levels complement the ICCS network to provide timing signals. The extended-range fast timing level (100-ns resolution and stability) has 250 channels; the fast timing level (1-ns resolution and stability) has 900 channels; and the precision timing level (30-ps resolution and stability) has 50 channels. (40-00-0298-0333pb01)

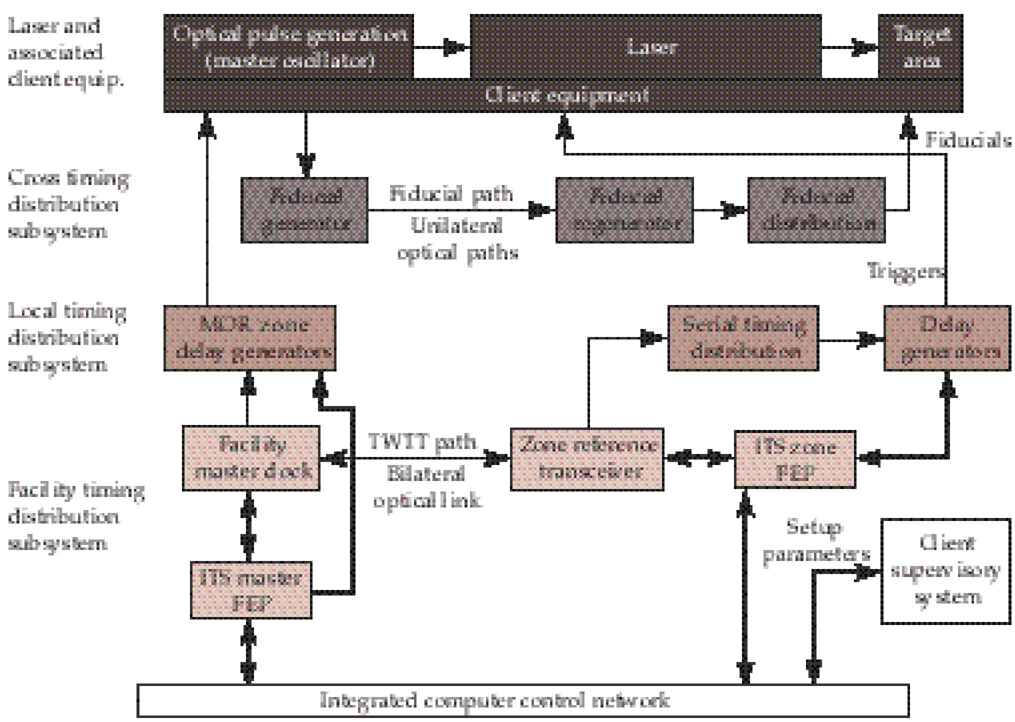


FIGURE 12. Three subsystems make up the integrated timing system, which supplies coordinated trigger and fiducial signals to NIF equipment. (40-00-0298-0334pb01)

(TWTT) which continuously monitors and compensates for transmission pathlength changes. TWTT will service 11 remote clocks, distributed throughout the Laser and Target Area Building (LTAB) (Figure 13). Each clock will serve a zone with several timing clients. For instance, one of the clocks in laser bay No. 1 will provide timing for preamplifier modules, energy diagnostics, power diagnostics, and imaging diagnostics. Within each zone, the local timing distribution can be easily expanded.

The broadcast serial encoded data—which travels from the zone master clock through the 1:8 splitters to the hex-delay generator—carries all the timing information needed, including epochs, gates, and a phase-critical reference clock. Epochs generate triggers at a fixed repetition rate. We have several specified epoch rates between 0.2 Hz and 960 Hz to support current needs, and have at least two others available for special needs and future growth. Gates generate single-shot triggers. Two gates are reserved for the shot director application—one for dry run / rod shots, the other for system shots. Clients can reserve other gates for use as diagnostic tools and so on. When a client is done, the gate is released for use by others. Finally, a high-frequency, phase-critical reference clock (155.52 MHz) is recovered in each delay generator.

The ITS software provides the link between client requests and generated triggers, and a number of databases provide information needed to operate and maintain the ITS. For instance, two databases are maintained by clients: the shot setup database, which contains client setup parameters, and the laser pathlength database, which keeps track of optical path lengths. The ITS-maintained databases are used to generate control bit patterns. For instance, the local t_0 database computes local t_0s for each client, based on contents of the laser pathlength database. The ITS timing path database contains data on internal ITS pathlengths to correct for detected changes and accelerate recovery due to hardware replacement.

The ITS and the master oscillator maintain a critical timing relationship; the master oscillator room maintains the fiducial in sync with the ITS reference, and the transmitted beams in sync with the fiducial. The ITS also provides optical and electrical fiducials to cross-time diagnostics.

Title II Activities

We have ordered a demonstration facility timing system consisting of a master clock and single remote clock; we anticipate the system will have a timing stability of ~200 ps. We will evaluate and enhance this system early in Title II to reach precision performance levels. A delay generator that will meet NIF precision performance goals is not currently available, but we are working with industry to modify existing designs to enhance stability and meet NIF goals. We also plan to be compatible with the French Laser Megajoule (LMJ) project timing system, so that each of us will have an alternate delay generator source.

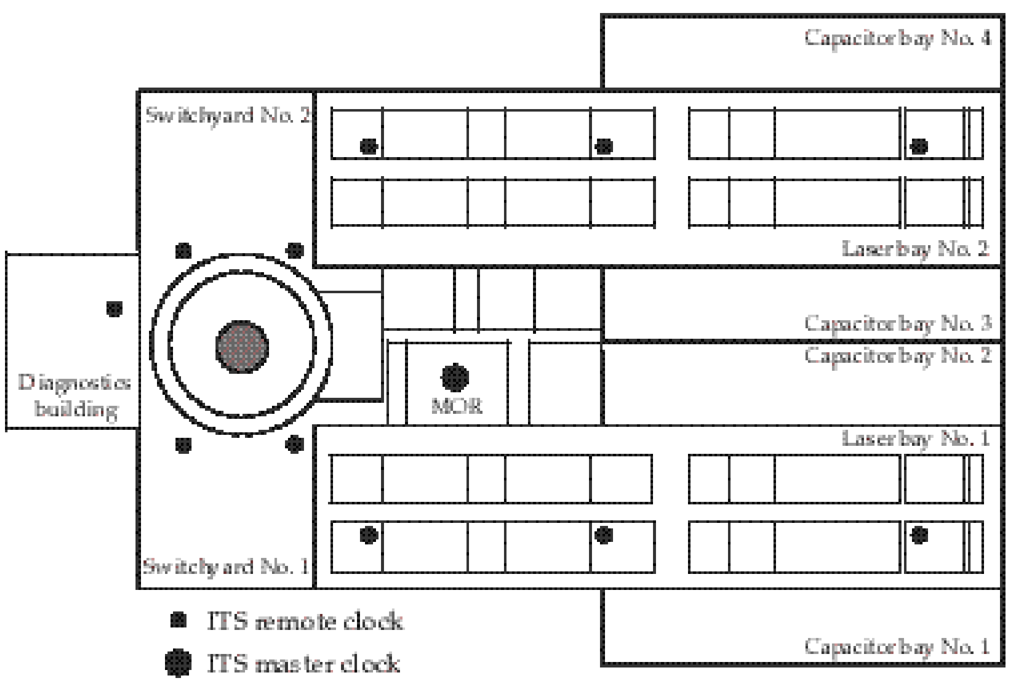


FIGURE 13. Location of the remote clocks and master clock in the NIF buildings. (40-00-0298-0335pb01)

Industrial Controls System

The industrial controls system (ICS) includes the industrial control FEPs, the facility environmental monitor, the communications system, and the integrated safety system. Figure 14 shows a block diagram of the ICS.

Industrial Controls FEP, Facility Environmental Monitor, and Communications System

The two ICS FEPs interface 12 separate industrial controls application areas to the supervisory system:

- The amplifier cooling system.
- The beam transport vacuum system.
- Beam transport gas system.
- Facility environmental monitor.
- Target chamber vacuum system.
- Environmental protection system.
- Tritium processing system.
- Personnel, safety, and occupational access.
- Final optics assembly (FOA) thermal control system.
- Safety interlock system.
- Access control system.
- t-1 abort system.

The last three systems, which form the integrated safety system, are discussed separately in the "Integrated Safety System" section (p. 211).

Programmable logic controllers (PLCs) individually control all but one of these application areas. (The access control system is controlled by a personal computer.) Communications between the ICS FEP and the various subsystems will occur over the ICCS Ethernet network, while communications to the supervisory layer will occur over the ICCS network via CORBA. The industrial controls FEP is scoped to be a SPARC 5 processor in a VME crate running the Solaris UNIX operating system. There are interfaces for control and monitoring, and others for monitoring only (see Table 1).

The environmental monitor must measure and display facility environmental parameters affecting the laser's performance, i.e., temperature, relative humidity, oxygen content, vacuum, argon flow, and nitrogen flow. It must also archive environmental parameters, provide trending displays and reports, and provide machine interlocks to prevent damage to equipment or loss of shot data due to improper operating conditions. This monitor acquires facility environmental data from network data acquisition modules located throughout the facility and acquires additional environmental data from the amplifier cooling system, the spatial filter

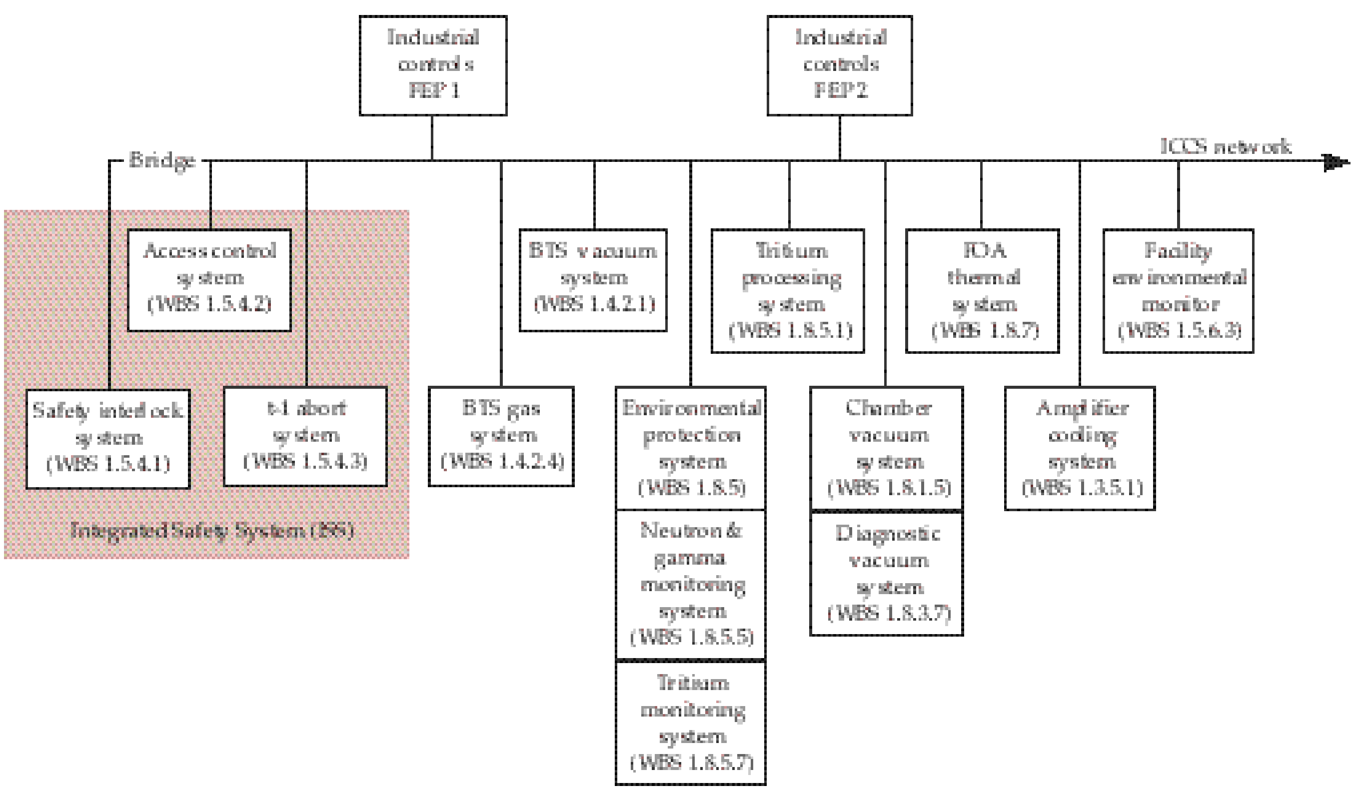


FIGURE 14. Block diagram of the industrial controls system (ICS). (40-00-0298-0336pb01)

TABLE 1. Industrial controls interfaces for control and monitoring, and for monitoring only.

Controlling and monitoring	Monitoring only
Beam transport vacuum control system	Facility enviornmental monitor(data acquired via network data acquisition modules)
Beam transport gas control system	Safety interlock system
Amplifier cooling system	Access control system
FOA thermal control system	t-1 abort system
Target chamber vacuum control system	Envionmental protection system

vacuum system, the target chamber vacuum system, and the gas control system. Figure 15 shows the type of sensors employed by the monitor and their location in the facility. The facility environmental monitor provides machine interlocks for situations such as too high a vacuum in the spatial filter, too high an oxygen content for the amplifiers, and out-of-range temperatures in the KDP crystals.

The communications system provides radio communications throughout the facility as well as a video surveillance system. Radio communications includes 50 hand-held transceivers and the necessary repeaters. The video surveillance system consists of 32 surveillance cameras, which may be manually selected for viewing or scanned sequentially. A time-lapse recording system records the camera images.

Title II Activities

For the industrial controls system, our high-priority Title II tasks are to finalize designs for the FEPs and facility environmental monitor, make the final sensor selections, complete enumeration of the machine interlocks, and complete final surveillance system design and camera location assignments.

Integrated Safety System

The integrated safety system (ISS) consists of the safety interlock system (SIS), the access control system (ACS) and the t-1 abort system.

The SIS requirements include providing and controlling the laser and target area status panels. It must determine the hazards from laser light, ionizing radiation, high voltage, and oxygen depletion, and provide and control audible alarms—such as klaxons and

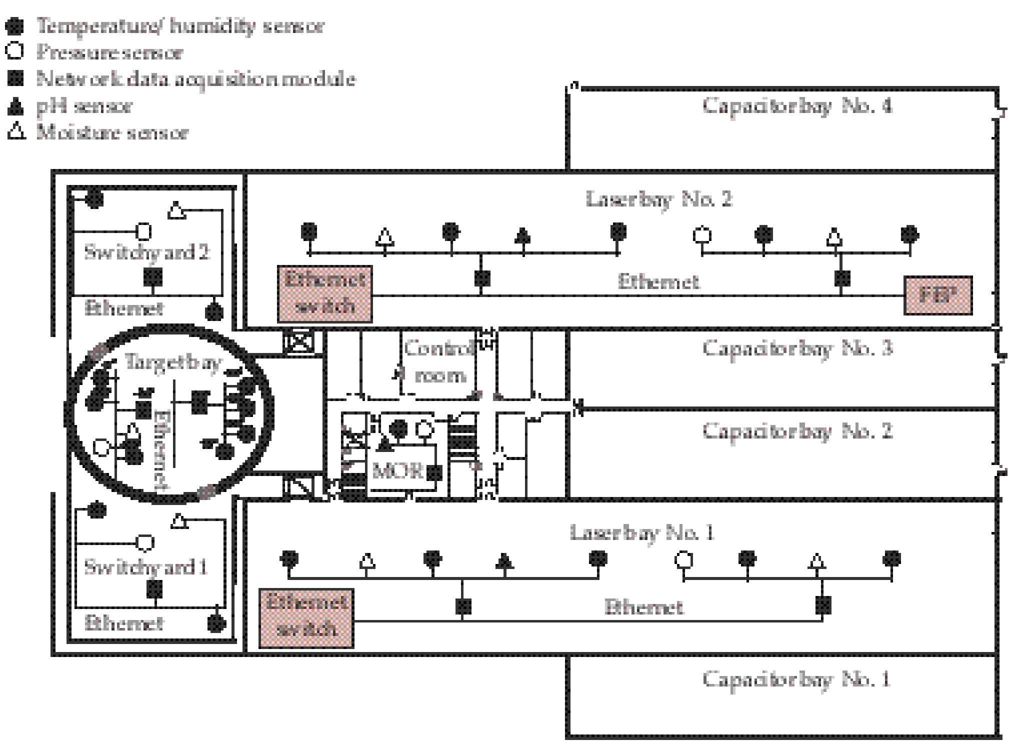


FIGURE 15. Sensor locations for the NIF facility environmental monitor.
(40-00-0298-0337pb01)

horns—as well as automatic voice annunciation of hazard-level changes. The system is also required to monitor door positions, crash buttons, and shutter positions, and provide permissives to power conditioning and safety shutters. The SIS must be modular to support partial system operation. It must also provide operator control and status screens, and must fail to a safe condition (be "fail-safe") if there is a loss of AC power.

The SIS distributed system is based on four PLCs. Three PLCs control interlocks in the major areas of the facility, while the fourth "master" PLC oversees the others. As required, the SIS will be capable of functioning in a stand-alone mode, apart from the industrial control FEP or the rest of the ICCS. The SIS is a "safety shutdown" system, not a "mitigating" system. That is, when an abnormal situation is detected, the SIS shuts down the affected system or systems, rather than trying to mitigate the hazard and continue operation. The SIS is designed to be fail-safe, in that it will (1) shut down if there is a communications loss or power loss, (2) allow unrestricted egress from the facility, (3) act as a "watchdog" for the timers that are part of the I/O drops, (4) provide feedback monitoring of all critical outputs, and (5) use internal diagnostics. Although the SIS does not perform process control functions, it does issue control "permissives" allowing interlocked systems to operate. The master PLC performs comparative error checking on the other PLCs in the system. We expect the scan time for the SIS to be about 70 to 100 ms. The SIS monitors and controls 28 "controlled" doors, 80 "monitored" doors, 125 run-safe boxes, 83 status display panels, 28 oxygen monitors, 18 radiation monitors, and 278 outputs (permissives, shutters, etc.). Figure 16 shows the safety devices in a typical NIF beamline; Figure 17 shows the approximate locations of selected devices in the facility.

The ACS must monitor entry and egress of all personnel through the major access points, use names and training records to identify personnel and approve their access, log all monitored access point transactions, and display a summary status. The ACS is designed to function with the SIS and an on-line database of qualified personnel to control access into the NIF. The system allows entry and egress through monitored doors by sensing special badges that will be carried by all personnel and visitors. All movement into, out of, and within the facility will be recorded into a transaction log available to the higher-level ICCS systems. As of Title I, we plan to base the ACS on a personal computer system. The ACS will be able to function as a stand-alone system, without intervention from the industrial controls FEP or other ICCS components. We plan to purchase the ACS as a "turn-key" system, based on a performance specification.

The t-1 abort system is required to work with the ITS to monitor selected components in each beamline before allowing a system shot. The t-1 abort system will abort a shot if the selected components in the active beamlines do not reach their shot configuration positions during the final seconds. We will also be able to configure the abort system to choose which beams to monitor. The components monitored during the last second of the countdown are (1) the output sensor, as 192 mirrors move into the beam path and 96 filter wheels move into position, (2) the input sensor, as 48 alignment laser mirrors move out of position [excluding the preamplifier beam transport system (PABTS) alignment laser and allowing the four-pass to the PABTS], and (3) the cavity spatial filter, as 192 wave plates move out of the beam path. At t-1, the system will begin monitoring the input and output sensors. As of Title I, the abort system is scoped as a PLC-based system.

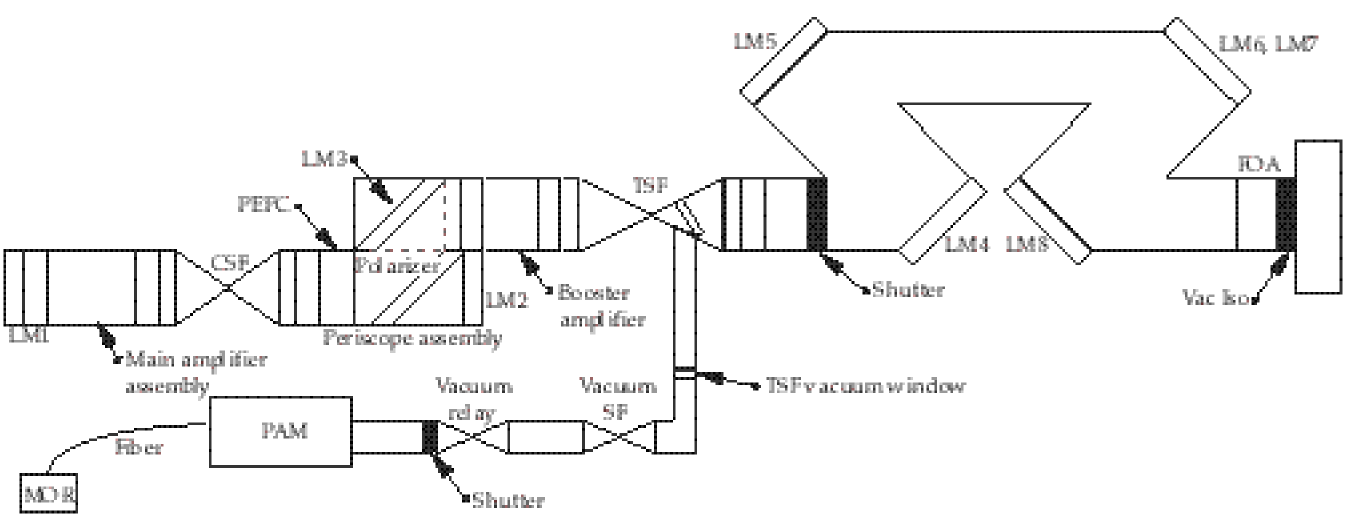


FIGURE 16. Location of safety devices (colored black) in a typical NIF beamline. (40-00-0298-0338pb01)

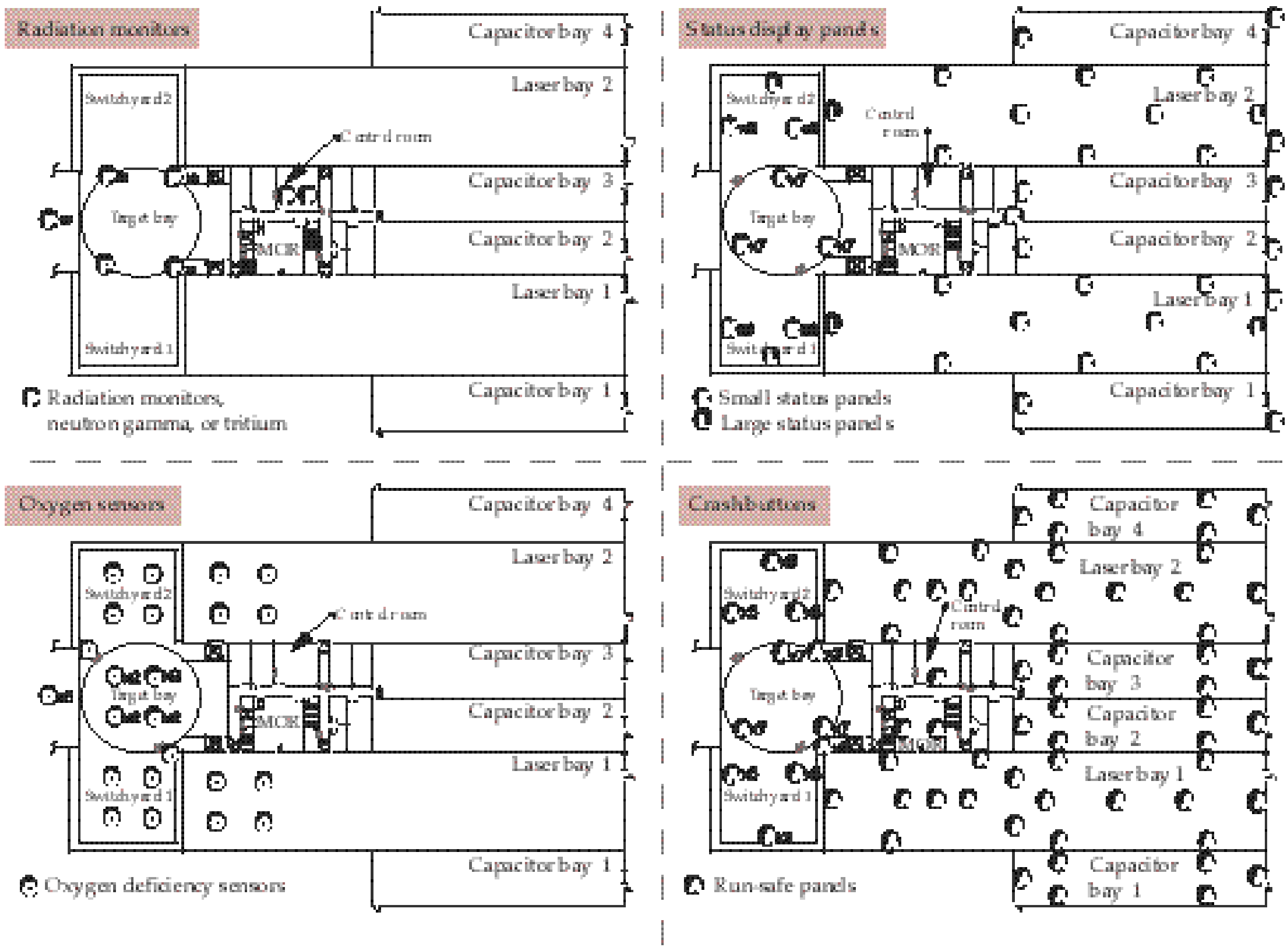


FIGURE 17. The approximate locations of radiation monitors, status display panels, oxygen sensors, and crash buttons for the NIF's safety interlock system (SIS). (40-00-0298-0339pb01)

Title II Activities

Our high-priority Title II tasks for the ISS are to select a PLC vendor, complete a detailed SIS I/O layout, complete the design of run-safe and status panels, complete design of the SIS interlock strings, complete ISS test plans, and complete ACS procurement specifications.

For more information, contact
Paul J. VanArsdall
Integrated Computer Control System Lead Engineer
Phone: (925) 422-4489
Email: vanarsdall1@llnl.gov
Fax: (925) 422-1930